AN ARGUS
SPECIALIST
PUBLICATION

YOUR Commodore

July 1984

How to track down and use your 64's hidden memory

Great games for the unexpanded VIC-20

Latest Commodore software star-rated for you

Be a hero with our Commodore 64 game

commodore
commodore VIC-20
FIRE ANT
COMM
TURTLE JUMP
64

FREE with Personal Computing Today

ANIROG
BONGO
Space Pilot
Flight Path 737

Bongo

Enjoy the hilarious antics of the comical mouse as he collects the lost diamonds. He climbs ladders, slides down chutes, use transporter and trampolines to jump across the divide. Multi screen game with three levels of difficulty.
Commodore 64 £7.95 — Vic 20 £7.95 — Spectrum £5.50

Space Pilot

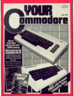
Realise your dreams of being king of the open skies. Fly your aircraft into unrelenting dog fights with enemy fighters. Prove how well you can handle your craft. Five stages of tough engagements.
Commodore 64 £7.95 — Spectrum £5.50

Flight Path 737

An advanced Pilot Trainer. Written by a flight simulator instructor and pilot. Panoramic Pilot's eye view.
Commodore 64 £7.95 — Vic 20 £7.95

Also available on Disk at £9.95

TRADE ENQUIRIES: ANIROG SOFTWARE LTD. 29 WEST HILL DARTFORD KENT (0322) 92513/8
MAIL ORDER: 8 HIGH STREET HORLEY SURREY 24 HOUR CREDIT CARD SALES HORLEY (02934) 6083
PAYMENT BY CHEQUE P.O. ACCESS/VISA 50p POSTAGE & PACKAGING £2.00 OVERSEAS



Editor
Paul Liptrot

Designer
Paul Hunter

Managing Editor
Ron Harris

Group Editor
Wendy Palmer

Advertisement Manager
Mike Segrue

Divisional Advertisement Manager
Beverley McNeill

Chief Executive
Jim Connell

Argus Specialist Publications Ltd, No. 1 Golden Square, London W1R 3AB.

Printed by The Passmore Print Group Ltd at St Albans, Herts. Trade Distribution: Argus Press Sales and Distribution Ltd, 12-14 Paul Street, London EC2A 4JS. 01-247 8233. Design and origination: MM Design, Circus House, 26 Little Portland Street, London W1N 5AF.

The contents of this publication, including all articles, plans, drawings and programs and all copyright and other intellectual property rights therein belong to Argus Specialist Publications Ltd. All rights conferred by the Law of Copyright and other intellectual property rights and by virtue of international copyright conventions are specifically reserved to Argus Specialist Publications Ltd and any reproduction requires the prior written consent of the company.

© Argus Specialist Publications Ltd

Contents

Your Commodore, a free supplement to Personal Computing Today, July 1984

VIC-20 programs 4
Two fun-to-play games for the unexpanded VIC by Mathew Solly. In one you're under siege and the second could make you rich...

CBM 64 program 6
Here's your chance to be a hero. Just type in Thomas Turnbull's program and save the potholers

CBM 64 programming.............................. 9
Track down and use the hidden memory in your 64. Your guide is Allen Webb

CBM 64 software reviews 13
Our experts give you their ratings for new games and some more serious software

VIC-20 program................14
Worms can make a tasty dish... if you're a bird. That's your role in Peter Godbehere's game

Two very different games for the unexpanded VIC-20 by Mathew Solly. In the first you're under siege and the second could make you rich . . .

Castle Siege

YOU ARE the last survivor in the castle and your only weapon is a large pile of rocks.

You are under attack from the enemy who are scaling the castle wall. You must drop the rocks to dislodge them. If they reach the top the game is over. How long will you survive?

The control keys are: **Z** left, **X** right and **space bar** to drop the rocks.

- Castle Siege is in two parts. Type in **Listing 1**, which defines the characters, RUN and then NEW. Now type in **Listing 2**. See also note at bottom.

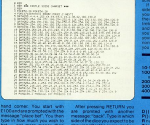
How it works

10-100 main game loop
1000-1500 initialize variables and strings
2000-2500 instructions
3000 POKE on other side of screen
3500 POKE on other side of screen
4000-4500 throw rock
5000-5500 PRINT enemy
6000-6500 game over
7000-7500 rock hit enemy routine
8000-8500 pick up rock routine

Variables

EH (), EL () climbing enemy
D defender
R rock
FR falling rock
RO, DE test for rock and defender
E number of enemies
S4 sound channel
CO colour location
T time
BT best time
W wall location
L, Q loops
TE () test for enemy at top

- **Listing 1 — defines characters for Castle Siege**



Crown and Anchor

THIS GAME is played with three dice which are displayed on the screen.

The amount of money you have is displayed in the top left hand corner. You start with £100 and are prompted with the message "place bet". You then type in how much you wish to bet — not more money than you have!

After pressing RETURN you are promted with another message: "back". Type in which side of the dice you expect to be showing after the computer has thrown them. The computer will then throw the dice and the outcome will be printed on the screen.

If one of the dice shows the side you backed you get double your stake money back. If two of the dice show your side you get treble your stake money back, and if all the dice show the side you backed you get four times your stake money returned. If none of them show your side you lose your stake.

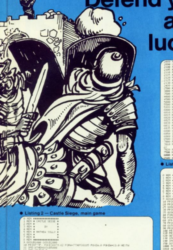
How it works

10-100 main game loop
1000-1500 initialize variables and strings
2000-2500 instructions
3000-3500 print credit
4000-4500 bleep

Variables

D () dice
P () position of numbers on dice
R number of dice right
T total credit
CO colour location

Defend your castle... and try your luck with dice



● **Listing 2 — Castle Siege, main game**

```
0 REM ****************
1 REM * CASTLE SIEGE *
2 REM *              *
3 REM *      BY      *
4 REM *              *
5 REM * MATHEW SOLLY *
6 REM *              *
7 REM *              *
8 REM ****************
9 GOSUB1000:GOSUB2000
10 PRINT"■":POKE36879,62:FORW=7790TO8185:POKEW,0:POKEW+CO,0:NEXTW
11 IFE<6THENGOSUB5000
12 IFE=6THENE=5
13 FORL=1TOE:PRINT"■■C.SIEGE■■■■TIME":T
14 IFDE=1THENPOKED,28:POKED+CO,1
15 IFRO=1THENPOKER,27:POKER+CO,2
16 IFDE=0THENPOKED,29:POKED+CO,1
17 IFRO=0THENFR=FR+22:POKEFR-22,0:POKEFR-22+CO,0:POKEFR,27:POKEFR+CO,2
18 IFFR>8185THENGOSUB8000
20 EH(L)=EH(L)-22:EL(L)=EL(L)-22
22 POKEEL(L),31:POKEEL(L)+22,0:POKEEH(L),30
23 IFEH(L)=TE(L)-396THEN6000
29 FORO=1TOE
30 IFFR=EH(O)ORFR=EL(O)THENGOSUB7000
31 NEXTO
90 GETK$:IFK$="Z"THENR=R-1:D=D-1:POKER+1,32:POKED+1,32
91 IFK$="X"THENR=R+1:D=D+1:POKER-1,32:POKED-1,32
92 IFK$=" "ANDRO=1THENGOSUB4000
93 IFD=7767THENGOSUB3000
94 IFD=7790THENGOSUB3500
97 T=T+1
98 IFBT<TTHENBT=T
99 NEXTL
100 E=E+1:GOTO11
1000 POKE36879,25:POKE36878,15:POKE36869,255:POKE650,255
1005 DIMEH(6):DIMEL(6):DIMTE(6)
1010 CO=30720:R=7756:D=7778:T=0:BT=0:RO=1:DE=1:E=1:S4=36877
1500 RETURN
1999 REM[CLR][CRSR RIGHT][RED](POUND)
2000 PRINT"■":PRINT"■■CASTLE SIEGE BY M.S."
2009 REM[CRSR RIGHT][BLK]
2010 PRINT"■■///////////////////"
2039 REM[BLU][CRSR DOWN][CRSR DOWN]
2040 PRINT"■YOU CONTROL THE LAST  ■REMAINING SURVIVOR OF ■THE CASTLE > YOUR ONLY
2049 REM[CRSR DOWN][CRSR DOWN]
2050 PRINT"REMAINING WEAPON IS A ■LARGE PILE OF ROCKS.  ■YOU MUST DROP ROCKS TO"
2059 REM[CRSR DOWN][CRSR DOWN]
2060 PRINT"DISLODGE THE CLIMBING ■ENEMY.THE GAME IS OVER■WHEN THE ENEMY REACHES"
2070 PRINT"THE TOP."
2075 REM[CRSR DOWN][5 CRSR RIGHT][PUR]
2080 PRINT"■■■■■PRESS ANY KEY."
2090 GETK$:IFK$=""THEN2090
2100 PRINT"■■■■■■THE KEYS ARE"
2105 REM[CLR][5 CRSR RIGHT][RED]
2110 PRINT"■■■■■Z.LEFT X.RIGHT"
2115 REM[CRSR DOWN][4 CRSR RIGHT][PUR]
2120 PRINT"■■■■    .DROP ROCKS"
2125 REM[4 CRSR DOWN][GRN]
2130 PRINT"■■■■PRESS ANY KEY TO BEGIN"
2499 GETK$:IFK$=""THEN2499
2500 RETURN
3000 POKER,32:POKED,32:R=7767:D=7789:RETURN
3500 POKER,32:POKED,32:R=7746:D=7768:POKE7790,0:RETURN
4000 POKER,32:POKED,29:RO=0:DE=0:LETFR=R+44
4500 RETURN
5000 TE(E)=INT(8164+22*RND(0)):EH(E)=TE(E):EL(E)=EH(E)+22
5500 RETURN
5999 REM[9 CRSR DOWN][6 CRSR RIGHT][RED]
6000 POKEEL(L),0:POKEEH(L),36:POKEEH(L)+CO,0:PRINT"■■■■■■■■■■■■■■■GAME OVER"
6010 FORL=1TO1000:NEXTL
6015 REM[CLR][4 CRSR DOWN][5 CRSR RIGHT]
6020 PRINT"■":POKE36879,25:PRINT"■■■■■■■■■ANOTHER GAME"
6025 REM[2 CRSR DOWN][9 CRSR RIGHT]
6030 PRINT"■■■■■■■■■■■Y/N."
6032 REM[6 CRSR DOWN][6 CRSR RIGHT][BLU]
6035 PRINT"■■■■■■■■■■■■■B.TIME":BT
6040 GETK$:IFK$=""THEN6040
6050 IFK$="Y"THENEH=1:DE=1:RO=1:E=1:T=0:H=0:L=0:GOTO10
6060 IFK$="N"THENPOKE36879,27:PRINT"■■":POKE36869,240:END
6500 GOTO6040
7000 POKEEL(O),0:POKEEH(O),35:POKEFR,35:POKES4,128
7010 FORL=1TO100:NEXTL:POKEEH(O),0:POKEFR,0:POKEFR+CO,0:EH(O)=TE(O):EL(O)=EH(O)+
22
7020 POKES4,0:GOSUB5000
7500 RETURN
8000 FR=0:RO=1:DE=1:POKED,43:FORO=1TO50:NEXTO
8500 RETURN
```

● **Listing 3 — Crown and Anchor**

```
0 REM ****************
1 REM *CROWN & ANCHOR*
2 REM *              *
3 REM *      BY      *
4 REM *              *
5 REM * MATHEW SOLLY *
6 REM *              *
7 REM *              *
8 REM ****************
9 GOSUB1000:GOSUB2000
10 PRINT"■■":GOSUB3000:PRINTA$;B$;B$;B$;B$;B$;C$
20 PRINT"■■■■■■■■■■■■■PLACE BET■■":INPUTB:IFB<2ORB>TTHEN20
25 GOSUB4000:T=T-B:GOSUB3000
30 PRINT"■■■■■■■■■■■■■■■■■■BACK■■":INPUTN:N=N+48:IFN<49ORN>54THEN30
35 GOSUB4000
40 FORL=1TO3:D(L)=INT(49+6*RND(0)):IFD(L)=NTHENR=R+1
50 POKEP(L),D(L):POKEP(L)+CO,1:NEXTL
60 IFR<>0THEN90
70 PRINT"■■■■■■■■BAD LUCK":FORL=1TO2000:NEXTL:IFT=0THEN100
80 GOTO10
90 PRINT"■■■■■■■■WELL DONE":FORL=1TO2000:NEXTL:T=T+B*(R+1):R=0:GOTO10
100 PRINT"■■■■■■■■■■■■■YOU'RE BROKE"
110 PRINT"■■■■■■■■PLAY AGAIN■■■■■■■■■■■■■■■■■■■Y/N>"
120 GETK$:IFK$=""THEN120
130 IFK$="Y"THENT=100:GOTO10
140 IFK$="N"THENPOKE36879,27:PRINT"■■":END
150 GOTO120
1000 POKE36879,8:POKE36878,15
1010 A$="■┌──┐  ┌──┐  ┌──┐"
1020 B$="│  │  │  │  │  │"
1030 C$="└──┘  └──┘  └──┘"
1040 DIMD(3):DIMP(3):R=0:T=100:P(1)=7914:P(2)=7922:P(3)=7930:CO=30720
1500 RETURN
2000 PRINT"■■■■■CROWN AND ANCHOR"
2010 PRINT"■■■▁▁▁▁▁▁▁▁▁▁▁▁▁▁▁▁▁▁"
2020 PRINT"■THIS GAME CONSISTS OF ■THREE DICE,YOU START  ■WITH A CREDIT OF \100■
"
2030 PRINT"TO GAMBLE ON THE OUT- ■COME OF THE DICE.■"
2040 PRINT"YOU MUST PUT YOUR BET ■ON ANY ONE OF THE      ■DICES SIDES.■"
2045 PRINT"■■■■■■PRESS ANY KEY"
2047 GETK$:IFK$=""THEN2047
2050 PRINT"■■THE COMPUTER THEN     ■THROWS THE DICE.IF ONE■OF THEM SHOWS THE SID
E"
2060 PRINT"YOU PICKED YOU GET    ■DOUBLE YOUR MONEY BACK■IF TWO SHOW YOUR SIDE■"
2070 PRINT"YOU GET TREBLE YOUR   ■MONEY BACK AND IF ALL ■THE DICE SHOW YOUR■"
2080 PRINT"SIDE YOU GET FOUR     ■TIMES YOUR BET BACK.■"
2090 PRINT"■■■■■PRESS ANY KEY";
2490 GETK$:IFK$=""THEN2490
2500 RETURN
3000 PRINT"■■CREDIT \          ■■■■■■■■■■■■";T;"■■"
3500 RETURN
4000 POKE36876,220:FORL=1TO100:NEXT:POKE36876,0:RETURN
```

We have inserted REMs in the lines above those in which Commodore control characters appear in the listings. These are only for your guidance and should not be entered. Remember that lines in some listings may not fit unless you use abbreviations for the BASIC keywords — you'll find them in your manual

Trap

Potholers are doomed unless you can get them to safety. Thomas Turnbull gives you the chance to be a hero — or end in a multi-coloured blast

oped!

When typing in this listing, you'll find REMs above lines containing control characters. These REMs are only for your guidance and should not be entered. And remember to abbreviate BASIC keywords — there's a list in your manual. Some lines may not fit without abbreviations. And it's quicker to type them in . . .

CAVERS are trapped deep underground — and they're relying on you and your rescue craft.

You must steer your way through the tunnel, avoiding the stalagmites and the sides, into a large cavern.

There you see the potholers signalling with a lamp, a flashing sprite which also changes shape due to a kind of BASIC interrupt.

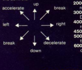
Touch the lamp with your craft and a bell rings, indicating the first part of your mission is complete. Now you and the rescued potholers must reach safety.

This time you use the right hand tunnel and touch your craft against a sprite with the word "winner" written inside it. This records your time score on the table.

Don't touch this sprite on the way in — it will be ignored until you have rescued the cavers.

Touching any other objects will result in a multi-coloured explosion, complete with the sound of the blast.

The game works with the joystick in port 2 and these are the controls:



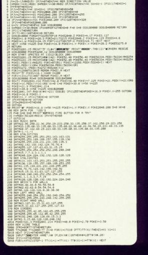
The fire button is used to restart the game and to give instant slow speed.

You will find that the response to the joystick is instant, even though the controls in lines 1 to 25 are written in BASIC.

As the Commodore 64 finds sub-routines by searching line numbers from zero, I placed this routine at the beginning so it finds the routine and acts on it quickly.

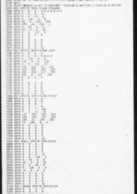
How it works

1-25 main joystick and control module
27-35 reset memory and store sprites and user-defined graphics
36-117 main section: switches on sprites, places them on screen and moves them
118-125 theme tune music DATA
130-410 user-defined graphics DATA
5000-5110 top 10 time table — a high-score table for the 10 best times
7000-8750 sprite DATA
9000-9015 bell sound effect
10000-10020 explosion effect
20000-25000 draw cavern routine
30000-30075 winner sprite DATA
40000-40100 win and display routine
45000-45150 instructions
50000-50070 routine to play theme music
60000 random high-resolution colour routine

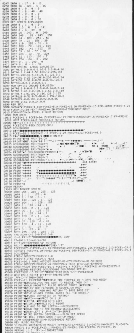


```
0 V=53248:S=54272:POKE54296,15:FORI=STOS+24:POKEI,0:NEXT:GOTO27
1 V=53248:POKEV+28,2:POKE53275,2:JV=PEEK(56320):IFSX=0THENSX=5
2 POKEV+28,2:POKEV+37,0:POKEV+38,4:FR=JVAND16:REM FIRE BUTTON STATUS
3 JV=15-(JVAND15):IFJV=0THENJV=K:REM DIRECTION VALUE
4 C=PEEK(V+31)AND1:D=PEEK(V+30)AND2:IFJV=10THENJV=K:SX=SX-1:IFSX<1THENSX=1
5 IFFR=0THENSX=2
6 IFJV=5THENJV=K:SX=SX+1:IFSX>50THENSX=50
7 IFJV=1THENY=Y-SX:POKE2040,200:IFY<0THENY=0
8 IFJV=2THENY=Y+SX:POKE2040,208:IFY>255THENY=255
9 IFJV=4THENX=X-SX:POKE2040,210:IFX<0THENX=0
10 IFJV=8THENX=X+SX:POKE2040,209:IFX>320THENX=320
11 IFJV=0THENPOKE2040,200
12 K=JV:IFD=2THENF=2:D=0:GOSUB9000
14 G=PEEK(V+30)AND8:IFG=8ANDF=2THEND=0:F=0:G=0:GOSUB9000:GOSUB40000:RETURN
19 IFC<>1THENRETURN
20 IF(TI-RO)<60THENC=0:RETURN
21 GOSUB10000:FORW=YTO225STEP10:POKE2040,Z:POKES+4,17:POKES,127
22 POKES+1,0:POKEV+28,3:FORZ=211TO215:POKE2040,Z:POKES+4,129:POKES+4,6
23 POKEV+1,W:POKES+24,15:FORT1=12TO0STEP-6:POKES+24,T1:NEXT:NEXT
24 NEXT:X=50:Y=225:D=0:POKEV+16,0:POKEV+1,Y:POKEV,X:POKEV+28,2:POKE53275,0
25 RETURN
27 POKE53281,15:PRINT"J":CL$="[illegible]":PRINT"[illegible]";TAB(13)"[illegible]CAVERN RESCUE
28 GOSUB9000:POKE53280,12:GOSUB50000:GOSUB9000
29 REM START SPRITES AT 13312(BLOCK 208)
30 POKE53281,15:PRINTCHR$(142):POKE52,48:POKE56,48:POKE56334,PEEK(56334)AND254
31 POKE53281,15:PRINTCHR$(142):POKE52,48:POKE56,48:POKE56334,PEEK(56334)AND254
32 POKE1,PEEK(1)AND251:FORI=0TO511:POKEI+12288,PEEK(53248+I):NEXT
33 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334)OR1
34 POKE53272,(PEEK(53272)AND240)+12
35 FORI=12000TO13015:READA:POKEI,A:NEXT
36 PRINT"J":POKEV+16,1:X=60:Y=200
37 FORI=13312TO13887:READA:POKEI,A:NEXT
38 GOSUB20000:RO=0:GOSUB40030:GOSUB45000
80 RO=TI:POKE2043,216:POKEV+42,0:POKEV+6,40:POKEV+7,225:POKEV+21,PEEK(V+21)OR8
81 POKEV+16,PEEK(V+16)OR8:XX=0:C=0:POKEV+39,0:X=50:Y=225
82 POKEV+21,9:POKE2041,214
83 POKEV+39,0:X=50:Y=225:GOSUB20000
84 POKE2041,INT(RND(0)*4)+211:GOSUB1:IFX>255THENPOKEV+16,9:POKEV,X-255:GOTO86
85 POKEV+16,8:POKEV,X
86 POKEV+1,Y:IFC=1THENC=0:GOTO90
87 IFK=5THENK=0:GOTO80
88 IFI=2ANDXX=5THEN90
89 GOTO84
90 PRINT"#":POKEV+16,8:X=50:Y=225:POKEV+1,Y:POKEV,X:POKE2040,208:D=0:XX=0
100 REM DATA FOR  "UDG'S
115 F=0:G=0:D=0:PRINT"[illegible]PRESS FIRE BUTTON FOR A TRY"
116 K=PEEK(56320)AND16:IFK=0THEN80
117 GOTO116
119 REM MUSIC DATA
120 DATA22,96,250,22,96,250,29,233,250,33,135,250,37,162,250,29,223,250
121 DATA33,135,60,33,135,60,33,135,60,22,96,60,22,96,50,29,223,60,33,135
122 DATA60,37,162,60,29,223,60,33,135,60,33,135,60,33,135,200
125 DATA-1,-1,-1
130 REM STALIOTITES
135 DATA255,255,255,253,253,163,163,131
140 DATA255,253,189,189,157,155,139,136
150 DATA255,255,239,231,231,199,135,3
162 DATA92,142,188,142,124,76,76,4
163 DATA255,127,127,63,63,31,15,15
164 DATA255,254,254,252,252,248,248,240
165 DATA15,7,7,3,3,1,1,1
166 DATA248,224,224,192,192,128,128,128
170 REM STALGIMITES
180 DATA131,163,163,253,253,255,255,255
190 DATA136,139,155,157,189,189,253,255
200 DATA3,135,199,231,231,239,255,255
212 DATA4,76,76,124,142,188,142,92
213 DATA15,15,31,63,63,127,127,255
214 DATA240,240,248,252,252,254,254,255
215 DATA1,1,1,3,3,7,7,15
216 DATA128,128,128,192,192,224,224,240
220 REM BRICK'S FOR WALLS
230 DATA46,46,46,0,58,58,58,0
260 DATA62,62,62,0,54,54,54,0
265 DATA0,126,126,126,0,30,30,30
270 REM LEFT HAND WALL
280 DATA255,240,254,252,192,248,254,252
290 DATA255,255,255,240,240,192,255,255
320 REM RIGHT HAND WALL
330 DATA63,127,31,15,31,63,127,255
340 DATA15,31,63,127,255,255,127,63
370 REM GIRDERS
380 DATA255,255,42,85,162,65,255,255
390 DATA255,255,65,162,85,42,255,255
400 DATA192,240,120,120,60,15,3,3
410 DATA3,3,15,60,120,120,240,192
2016 POKEV+21,3:POKE2041,214:POKEV+40,0:POKEV+2,70:POKEV+3,50
4999 REM SCORE TABLE
5000 IFRO=0ORTT<0THENRETURN
5005 T7=2000:TT=2000-TT:Z1=0:FORX1=1TO10:IFTT>TT(X1)THENZ1=X1:X1=11
5010 NEXT:IFZ1=0THENRETURN
5020 INPUT"[illegible]ENTER NAME";N$:IFLEN(N$)>20THENN$=LEFT$(N$,20)
5040 IFZ1=10THEN5060
5050 FORX1=9TOZ1STEP-1:TT(X1+1)=TT(X1):TT$(X1+1)=TT$(X1):NEXT
```

Will you survive to rescue the cavers?



```
5060 TT(21)=TT:TT$(21)=N$
5080 PRINT"J"TAB(6)"WTIME TABLE (SECONDS)"
5090 FORX1=1TO10
5095 PRINT"W":X1;TAB(10)INT((T7-TT(X1))*100)/100.TAB(23).TT$(X1)
5100 NEXT
5110 PRINT"WPRESS A KEY TO RESTART":POKE198.0:WAIT198.1:POKE198.0:RETURN
6999 REM SPRITE DATA PLANE FORWARD
7000 DATA 0 . 0 . 0 . 0.0.0.0.0.0.0
7005 DATA 0 . 0 . 0 . 36
7010 DATA 0 . 0 . 24 . 0
7015 DATA 0 . 126 . 0 . 0
7020 DATA 66 . 0 . 255 . 255
7025 DATA 255 . 64 . 126 . 2
7030 DATA 0 . 60 . 0 . 0
7035 DATA 60 . 0 . 0 . 66
7040 DATA 0 . 0 . 129 . 0
7045 DATA 0 . 0 . 0 . 0
7055 DATA 0 . 0
7060 DATA 0 . 0 . 0 . 0
7065 DATA 0 . 0 . 0 . 0
7070 DATA 0 . 0 . 0 . 0
7075 DATA 0 . 0 . 0 . 32
7099 REM SPRITE DATA PLANE LEFT
7100 DATA 0 . 0 . 0 . 0
7105 DATA 0 . 0 . 0 . 0.0.0.0
7110 DATA 0 . 128 . 0 . 0
7115 DATA 128 . 0 . 0 . 128
7120 DATA 0 . 0 . 192 . 15
7125 DATA 192 . 224 . 72 . 126
7130 DATA 255 . 255 . 255 . 255
7135 DATA 255 . 254 . 127 . 255
7140 DATA 252 . 0 . 127 . 240
7145 DATA 0 . 0 . 0 . 0
7150 DATA 0
7155 DATA 0 . 0 . 0 . 0
7160 DATA 0 . 0 . 0 . 0
7165 DATA 0 . 0 . 0 . 0
7170 DATA 0 . 0 . 0 . 0
7175 DATA 0 . 0 . 0 . 32
7200 REM SPRITE DATA PLANE RIGHT
7400 DATA 0 . 0 . 0 . 0
7405 DATA 0 . 0 . 0 . 0
7410 DATA 0 . 0 . 0 . 1
7415 DATA 0 . 0 . 1 . 0
7420 DATA 0 . 3 . 7 . 192
7425 DATA 7 . 4 . 72 . 15
7430 DATA 127 . 255 . 255 . 255
7435 DATA 255 . 255 . 127 . 255
7440 DATA 255 . 63 . 255 . 254
7445 DATA 15 . 240 . 0 . 0
7450 DATA 0 . 0 . 0 . 0
7455 DATA 0 . 0 . 0 . 0
7460 DATA 0 . 0 . 0 . 0
7465 DATA 0 . 0 . 0 . 0
7470 DATA 0 . 0 . 0 . 0
7475 DATA 0 . 0 . 0 . 32
7999 REM SMALL SPRITE EXPLOSION
8000 DATA 0 . 0 . 0 . 0
8005 DATA 0 . 0 . 0 . 0
8010 DATA 0 . 0 . 0 . 0
8015 DATA 0 . 0 . 0 . 0
8020 DATA 0 . 0 . 0 . 16
8025 DATA 0 . 0 . 130 . 0
8030 DATA 0 . 84 . 0 . 0
8035 DATA 40 . 0 . 0 . 56
8040 DATA 0 . 0 . 84 . 0
8045 DATA 0 . 130 . 0 . 0
8050 DATA 16 . 0 . 0 . 0
8055 DATA 0 . 0 . 0 . 0
8060 DATA 0 . 0 . 0 . 0
8065 DATA 0 . 0 . 0 . 0
8070 DATA 0 . 0 . 0 . 0
8075 DATA 0 . 0 . 0 . 32
8199 REM LARGER SPRITE EXPLOSION
8200 DATA 0 . 0 . 0 . 0
8205 DATA 0 . 0 . 0 . 0
8210 DATA 12 . 56 . 16 . 0
8215 DATA 1 . 0 . 128 . 0
8220 DATA 145 . 0 . 0 . 82
8225 DATA 0 . 0 . 36 . 0
8230 DATA 55 . 131 . 216 . 0
8235 DATA 0 . 0 . 0 . 20
8240 DATA 0 . 0 . 146 . 0
```



```
8245 DATA 1 . 17 . 0 . 2
8250 DATA 16 . 128 . 4 . 16
8255 DATA 0 . 0 . 0 . 0
8260 DATA 0 . 0 . 40 . 40
8265 DATA 0 . 0 . 0 . 0
8270 DATA 0 . 0 . 0 . 0
8275 DATA 0 . 0 . 0 . 32
8399 REM SPRITE EXPLOSION
8400 DATA 0 . 1 . 0 . 0
8405 DATA 113 . 0 . 1 . 141
8410 DATA 0 . 6 . 39 . 0
8415 DATA 28 . 243 . 0 . 249
8420 DATA 240 . 128 . 243 . 252
8425 DATA 64 . 103 . 251 . 56
8430 DATA 79 . 23 . 156 . 30
8435 DATA 110 . 220 . 31 . 6
8440 DATA 188 . 78 . 101 . 104
8445 DATA 102 . 141 . 112 . 18
8450 DATA 59 . 96 . 57 . 30
8455 DATA 224 . 13 . 70 . 224
8460 DATA 12 . 113 . 192 . 7
8465 DATA 254 . 64 . 3 . 252
8470 DATA 0 . 1 . 240 . 0
8475 DATA 0 . 0 . 0 . 32
8599 REM SPRITE  LARGE EXPLOSION
8600 DATA0.0.0.0.0.0.0.16.0.0.0.0.0.4.16
8610 DATA0.2.2.64.1.56.128.12.255.144
8620 DATA1.238.40.5.151.0.11.121.0.1
8630 DATA183.0.25.214.96.0.236.40.6.24
8640 DATA152.3.90.0.0.51.0.0.96.128.0
8650 DATA64.0.0.0.0.0.0.0.0.0
8699 REM SPRITE GROUND EXPLOSION
8700 DATA0.0.0.0.0.0.0.0.0.0.24.0.0.24
8710 DATA0.0.48.0.0.170.0.0.60.0.0.110
8720 DATA0.4.176.192.0.166.128.3.185
8730 DATA128.0.51.0.5.60.0.1.154.128.2
8740 DATA124.0.0.186.0.0.92.0.0.56
8750 DATA0.0.60.0.0.24.0.0
8999 REM BELL
9000 S=54272:POKES+1.130:POKES+5.9:POKES+15.30:POKES+24.15:FORL=0TO1:POKES+4.21
9010 FORU=1TO150:NEXT:POKES+4.20:FORU1=1TO20:NEXT:NEXT
9015 FORU=STOS+24:POKEU.0:NEXT:RETURN
10000 REM BANG
10010 POKES+1.6:POKES+24.15:POKES+4.129:FORT=15TO0STEP-.5:POKES+24.T:FF=FRE(0)
10020 NEXT:POKES+24.0:POKES+4.0:RETURN
20000 A$="J[illegible]"
20003 POKE53270.PEEK(53270)OR16
20005 GOSUB60000
20010 B$="[illegible]"
20013 POKEV+28.2:POKEV+21.11:POKEV+2.25:POKEV+3.63:POKEV+40.0
20015 C$="[illegible]"
20020 PRINT"@"A$:PRINTB$.C$
20030 PRINT"[illegible]":A1$+"[illegible]"
20035 GOSUB60000:PRINTA1$+"[illegible]"
20040 GOSUB60000:PRINTA1$+"[illegible]"
20050 GOSUB60000:PRINTA1$+"[illegible]"
20054 PRINT"[illegible]":GOSUB60000:PRINTA1$+"[illegible]"
20055 GOSUB60000:PRINTA1$+"[illegible]"
20060 GOSUB60000:PRINTA1$="[illegible]"
20070 GOSUB60000:PRINTA1$+"[illegible]"
20080 PRINT"[illegible]"
20090 PRINT"[illegible]"
20100 PRINT"[illegible]"
20110 PRINT"[illegible]"
20120 PRINT"[illegible]"
20130 PRINT"[illegible]"
20140 PRINT"[illegible]"
20160 PRINT"[illegible]"
20170 PRINT"[illegible]"
25000 RETURN
29999 REM BANNER SPRITE
30000 DATA 255 . 255 . 255 . 128
30005 DATA 0 . 1 . 170 . 0
30010 DATA 1 . 170 . 0 . 1
30015 DATA 140 . 128 . 1 . 128
30020 DATA 164 . 1 . 128 . 180
30025 DATA 1 . 128 . 44 . 1
30030 DATA 128 . 0 . 1 . 128
30035 DATA 36 . 1 . 128 . 52
30040 DATA 1 . 128 . 44 . 1
30045 DATA 128 . 0 . 1 . 128
30050 DATA 7 . 61 . 128 . 4
30055 DATA 37 . 128 . 6 . 61
30060 DATA 128 . 4 . 41 . 128
30065 DATA 7 . 37 . 128 . 0
30070 DATA 3 . 128 . 0 . 1
30075 DATA 255 . 255 . 255 . 32
40000 X=5:XD=0:C=0:D=0:F=0:G=0
40005 XX=0:PRINT"[illegible]YOU WIN"
40010 TT=INT((TI-AU)/6)/10
40015 IFTT<20THENPRINT"J":RETURN
40020 PRINT"[illegible]TIME=".TT
40030 POKE53275.7:POKEV+21.13:POKE2040.209:POKE2042.216:POKE2041.213:POKEV+39.0
40040 XX=XX+1:POKEV+4.30:POKEV.80:POKEV+1.100:POKEV+5.100:POKEV+41.1:POKEV+29.4
40045 GOSUB9000
40050 FORO=100TO255:POKEV+16.0
40060 POKEV+4.O-50:POKEV.O:NEXT
40070 POKEV+16.1:FORO1=0TO305:POKEV.O1-255:POKEV+4.O1-50:NEXT
40080 FORO=100TO225:POKEV+1.O:POKEV+5.O:NEXT:IFXX<>2THEN40040
40090 X=50:Y=225:D=0:POKEV+16.0:POKEV+1.Y:POKEV.X:POKEV+21.0:POKE53275.0
40100 GOSUB5000:RESTORE:GOSUB50000:GOSUB9000:RETURN
45000 POKE53281.15:PRINT"J[illegible]INSTRUCTIONS Y/N":POKE198.0
45010 WAIT198.1:GETF$:IFF$="N"THENRETURN
45020 IFF$<>"Y"THEN45000
45030 POKEV+21.13:PRINT"J[illegible]PEOPLE ARE TRAPPED IN A CAVE AND NEED"
45040 PRINT"[illegible]RESCUE.YOU ARE SENT TO RESCUE THEM IN"
45050 PRINT"[illegible]YOUR MAGNETIC PULSE RESCUE CRAFT ([illegible]MPRC[illegible])"
45055 PRINT"[illegible]THEY ARE FLASHING A SIGNAL.GO AND"
45060 PRINT"[illegible]RESCUE THEM AND RETURN TO BASE.BASE IS"
45070 PRINT"[illegible]THE WINNER SYMBAL.AVOID ALL THE ROCKS."
45080 PRINT"[illegible]PLACE JOY-STICK IN PORT#2"
45090 PRINT"[illegible]UP[illegible]-----[illegible]MPRC[illegible] GO'S UP
45100 PRINT"[illegible]LEFT[illegible]--[illegible]MPRC[illegible] GO'S LEFT
45110 PRINT"[illegible]RIGHT[illegible]-[illegible]MPRC[illegible] GO'S RIGHT
45120 PRINT"[illegible]UP AND TO THE LEFT[illegible]----ACCELERATE
45130 PRINT"[illegible]DOWN AND TO THE RIGHT[illegible]-DECCELERATE
45135 PRINT"[illegible]DOWN-LEFT & UP-RIGHT[illegible]--BRAKE
45140 PRINT"[illegible]FIRE BUTTON GIVES[illegible]-----SLOW SET SPEED
45145 PRINT"[illegible]A BELL RINGS WHEN PICK-UP MADE"
45150 XD=0:GOSUB40030:XD=0:GOSUB40030:RETURN
50000 REM MUSIC ROUTINE
50010 V1=54296:W1=54276:A1=54277:HF=54273:LF=54272:S1=54278:PH=54275:PL=54274
50020 POKEV1.111:POKELF+23.1:POKEW1.65:POKEA1.190:POKEPH.15:POKEPL.15
50030 READH1.L1.D1:IFH1=-1THENRETURN
50040 POKEHF.H1:POKELF.L1
50050 FORX1=D1-50TOD1-20:POKES1.136:NEXT
50060 FORT1=1TOD1:NEXT:POKEHF.0:POKELF.0:POKEW1.0
50070 GOTO50020
60000 A1$=MID$(CL$.INT(RND(0)*7)+1.1):RETURN
```

Find and use your 64's hidden memory

There's much more RAM than you might have thought tucked away in the Commodore 64. Allen Webb shows where it is and how to make it work for you

ONE OF the aspects which separates the Commodore 64 from the "also rans" is its large memory. While it is true that only about 38K is accessible from BASIC, with the aid of a few short routines, you will have ready access to around 60K.

First Let me ask some questions:

- Are you fed up with converting your machine code routines or blocks of data into lengthy BASIC loaders (similar to listing 2)?
- Are you interested in simple animation or do you wish to have access to several screens of data?
- Do you wish that you had a few kilobytes of protected data area for your adventure or simulation?
- Do you want to move blocks of data around rapidly and easily?
- Are you just an enthusiastic dabbler?

If you answer yes to any of these questions then read on.

Before launching into description of the routines, it is necessary to disducss how RAM is organised and controlled on the 64.

Consider **Figure 1.** The only obvious parts of RAM available for use are the BASIC area and the spare area. So where is the rest of the RAM?

The answer: hiding behind the ROMs and the I/O areas. Any address in these areas is shared by both RAM and ROM or I/O. The crunch is that a value POKEd to an address will be put into the RAM but a PEEK will reflect the value in ROM, not RAM.

Fortunately, the 6510 processor has an input/output control register at location 1. This location controls a whole handful of functions, as **Figure 2** indicates. By setting the correct bit to zero, the ROM area controlled by that bit will be switched out and the RAM will become available for use. If the bit is set, the ROM is switched back in.

hexadecimal	decimal	size of RAM	usage
\$0000-\$03FF	0-1024	1024	system
\$0400-\$07FF	1024-2047	1024	video memory
\$0800-\$9FFF	2048-40959	38912	BASIC area
\$A000-\$BFFF	40960-49151	8192	BASIC ROM
\$C000-\$CFFF	49152-53247	4096	spare RAM
\$D000-\$DFFF	53248-57343	4096	I/O colour RAM
\$E000-\$FFFF	57344-65535	8192	Kernal ROM

● **Figure 1 — Commodore 64 general memory map**

bit	function
0	switch for basic ROM
1	switch for Kernal ROM
2	switch for I/O area
3	cassette write line
4	cassette switch sense
5	cassette motor control

● **Figure 2 — the function of location 1**

Warning: Any attempt to switch out ROM by POKEing values into location 1 from BASIC will cause the machine to crash.

The routines described here use the switching out of ROMs to give you easy access to about 58K from BASIC.

Listing 1 gives the source code for the routines. I've included this since you may prefer to extract portions or modify it to suit your own purposes.

The first routine is called Blocksave. This routine will SAVE a specified block of memory to cassette or disc. Due to certain problems with saving with the interrrupts disabled, the routine will not SAVE the block behind \$D000 to \$FFFF. The syntax is simple:

SYS 49152 "filename", device,sa,start address,end address

commodore
CTRL
RUN STOP
SHIFT LOCK
SHIFT
BLK
WHT
RED
CYN
PUR
GRN
BLU
YEL

where Device is 1 for cassette or 8 for disc, sa = 2

As an example, to SAVE the BASIC ROM to disc use:
SYS 49152 "BASIC ROM",8,2,10*4096,12*4086-1

Location 1000 is used as a flag to determine whether you want to save the RAM under the BASIC ROM or not. A zero value will leave the ROM alone, a non-zero value will switch the ROM out.

Hence, if you precede the above example with POKE1000,1 then you will SAVE the RAM under the ROM, not the ROM. You will, however, get a LOAD error when you reload the saved RAM. Unless you have problems with your cassette, you can ignore the error.

Note: Any programs saved with blocksave must be loaded using the command:

LOAD "",device,1

otherwise it will not LOAD into the correct place.

As mentioned earlier, you cannot PEEK the ROM areas. The next routine, named Peekall, will do this job. The syntax is:

SYS 49155,address

the contents of the address will be returned in location 998. This routine will work on all areas.

The RAM under the ROMs offers great possibilities as virtual storage for animation or databases. The next command, Blockmove, is included to assist such applications. Quite simply, blockmove will take a specified slab of memory contents and put it at a specified location. The syntax is:

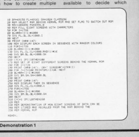
SYS 39158,sa,fa,da

where sa is the start address of the block

fa is the finish address of the block

da is the destination address

Demonstrations 1 and **2** use this command to show you how to create multiple screens for data or animation. As with blocksave, a flag is available to decide which

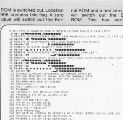


```
10 BM=49158:FL=49161:SN=1024:CL=55296
20 REM SELECT RAM BEHIND KERNAL ROM AND SET FLAG TO SWITCH OUT ROM
30 RA=14*4096:POKE996,0
40 REM FILL EIGHT SCREENS WITH CHARACTERS
50 FOR I=1TO8
60 BL=RA+(I-1)*1000
70 SYS FL,BL,BL+1000,I
80 NEXT
90 PRINT CHR$(147)
100 REM DISPLAY EACH SCREEN IN SEQUENCE WITH RANDOM COLOURS
110 FORI=1TO8
120 BL=RA+(I-1)*1000
130 SYS FL,CL,CL+1000,RND(1)*16
140 SYS BM,BL,BL+1000,SN
150 NEXT
160 C=C+1:IFC<10THEN100
170 REM SET UP EIGHT DIFFERENT SCREENS BEHIND THE KERNAL ROM
180 FOR I=1TO8
190 PRINT CHR$(147);O$=" SCREEN#"+STR$(I)
200 FOR J=1TO800:PRINTSPC(I)O$;:NEXT
210 BL=RA+(I-1)*1000
220 SYS BM,SN,SN+1000,BL
230 NEXT
240 PRINT CHR$(147)
250 REM DISPLAY THEM IN SEQUENCE
260 SYS FL,CL,CL+1000,1
270 FORI=1TO8
280 BL=RA+(I-1)*1000
290 SYS BM,BL,BL+1000,SN
300 NEXT
310 C=C+1:IFC<20THEN260
320 REM
330 REM DEMONSTRATION OF HOW EIGHT SCREENS OF DATA CAN BE
340 REM STORED AND RECALLED FROM THE RAM BEHIND THE
350 REM KERNAL ROM

READY.
```

● **Demonstration 1**

ROM is switched out. Location 996 contains this flag. A zero value will switch out the Kernal ROM and a non zero value will switch out the BASIC ROM. This has particular implications if you plan to move data to the colour RAM.



```
5 REM [RVS ON][WHT][8 CRSR RIGHT][BLK][CRSR DOWN][10 CRSR LEFT]
10 A$="[illegible]"
15 REM[RVS ON][COM+9][RVS OFF][7 CRSR RIGHT][BLK][CRSR DOWN][10 CRSR LEFT]
20 A$=A$+"[illegible]"
25 REM[RVS ON][COM+5][RVS OFF][CRSR DOWN][13 CRSR RIGHT]
30 A$=A$+"[illegible]"
35 REM[RVS ON][CRSR DOWN][17 CRSR LEFT]
40 A$=A$+"[illegible]"
45 REM[RVS ON][COM+5][RVS OFF][CRSR DOWN][13 CRSR LEFT]
50 A$=A$+"[illegible]"
55 REM[RVS ON][WHT][RVS OFF][8 CRSR RIGHT][BLK][CRSR DOWN][13 CRSR LEFT]
60 B$="[illegible]"
70 B$=B$+"[illegible]"
80 B$=B$+"[illegible]"
90 B$=B$+"[illegible]"
100 B$=B$+"[illegible]"
110 SC=1024:CL=55296:SB=14*4096:CB=10*4096:BM=49158
120 REM USE AREA BEHIND KERNAL ROM FOR GRAPHICS DATA
130 REM USE AREA BEHIND BASIC ROM FOR COLOUR DATA
140 REM DRAW AND SAVE 24 SCREENS
150 FORI=0TO12
160 PRINT CHR$(147)TAB(I*2)A$
170 BS=400*I
180 POKE996,0: SYS BM,SC,SC+200,SB+BS
190 POKE996,1:SYS BM,CL,CL+200,CB+BS
200 BS=BS+200
210 PRINT"[illegible]"TAB(I*2+1)B$
220 SYS BM,SC,SC+200,SB+BS
230 SYS BM,CL,CL+200,CB+BS
240 NEXT:PRINT CHR$(147)
250 BS=0:FORI=0TO22
260 REM DISPLAY SCREENS IN SEQUENCE AT A SPEED DETERMINED BY LINE 290
270 POKE996,0:SYS BM,SB+BS,SB+BS+200,SC
280 POKE996,1:SYS BM,CB+BS,CB+BS+200,CL
290 FORD=1TO30:NEXT
300 BS=BS+200:NEXT
310 GOTO250

READY.
```

- **Demonstration 2**

If you use the RAM behind the Kernal ROM for this purpose, you cannot move it to the colour RAM since both are switched out simultaneously. Demonstration 2 keeps the colour data behind the BASIC ROM to prevent this problem.

The next command, blockfill, fills a specified area with a specified character. This is useful for zeroing a block of RAM or filling the colour RAM with a specified colour. The syntax is:

SYS 49161,sa,fa,ch
where sa is the start of the block affected
fa is the finish of the block
ch is the character

The final command, charmove, is a specified form of blockmove. This command will download the normal upper case set of characters (256 characters in all) to a specified address. This will be of value if you plan to redesign your character set. The syntax is:

SYS 491, start address of destination.

All addressesare to be input as decimal values. All Parameters can be input as values, variables or expressions.

Listing 2 gives a BASIC loader for the routines. Once you have typed it in, SAVE it, since, while a crude checksum is included, you could make an error. Once it is up and running, why not save it using Blocksave? Like this:

SYS 49152 "name",dev,sa, 49152,49527

Demonstrations 1 and 2 give an idea of the possibilities of these routines.

Try them and enjoy.

```
1 !
2 !
10 TEMPSTORE     =  999
20 PEEKVAL       =  998
30 SAVEFLAG      =  1000
40 CHRCOM        =  $AEFD
50 FRMNUM        =  $AD8A
60 GETADR        =  $B7F7
70 SAVER         =  $E15F
80 SLPARA        =  $E1D4
90 *=$C000
100 !
110 !
120 J1            JMP BLOCKSAVE              ! SYS 49152 "NAME",DEVICE,SA,START
ADDRESS,FINISH ADDRESS
130 ! DEVICE...1 FOR CASSETTE, 8 FOR DISK22 ! SA...2
140 ! POKE 1000,0...ASSUMES NORMAL RAM
150 ! POKE 1000,1...SWITCHES OUT BASIC ROM
160 !
170 !
180 J2            JMP PEEKALL
190 ! SYS 49155,ADDRESS...PEEKS ANY LOCATION
200 !
210 !
220 J3            JMP BLOCKMOVE
230 ! SYS 49158, START OF BLOCK, END OF BLOCK, START OF DESTINATION
240 !
250 !
260 J4            JMP BLOCKFILL
270 ! SYS 49161, START OF BLOCK, END OF BLOCK, CHARACTER
280 !
290 !
300 J5            JMP CHARMOVE
310 ! SYS 49164, START OF DESTINATION
320 !
330 !
340 BLOCKSAVE     JSR SLPARA
350               JSR PARAMETER
360               LDA $14
370               PHA
380               LDA $15
390               PHA
400               JSR PARAMETER
410               LDX $14
420               LDY $15
430               PLA
440               STA $15
450               PLA
460               STA $14
470               LDA SAVEFLAG
480               BEQ CONT
490               LDA #54
500               STA $01
510 CONT          LDA #$14
520               JSR SAVER
530               LDA #55
540               STA $01
550               RTS
560 !
570 !
580 PEEKALL       JSR PARAMETER
590               LDA $14
600               STA $FB
610               LDA $15
620               STA $FC
630               LDY #0
640               LDA $01
650               STA TEMPSTORE
660               AND #240
670               SEI
680               STA $01
690               LDA ($FB),Y
700               STA PEEKVAL
710               LDA TEMPSTORE
720               STA $01
730               CLI
740               RTS
750 !
760 !
770 BLOCKMOVE     JSR PARAMETER              !START
780               LDA $14
790               STA $FB
800               LDA $15
810               STA $FC
820               JSR PARAMETER              !END
830               LDA $14
840               STA 990
850               LDA $15
860               STA 991
870               JSR PARAMETER              !DESTINATION
880               LDA $14
890               STA $FD
900               LDA $15
910               STA $FE
920               LDA $01
930               STA TEMPSTORE
940               AND #240
950               SEI
960               STA $01
970               LDY #0
980 AGAIN         LDA ($FB),Y
990               STA ($FD),Y
1000              LDA $FB
1010              CLC
1020              ADC #1
1030              STA $FB
1040              LDA $FC
1050              ADC #0
1060              STA $FC
1070              LDA $FD
1080              CLC
1090              ADC #1
1100              STA $FD
1110              LDA $FE
1120              ADC #0
1130              STA $FE
1140              LDA $FB
1150              CMP 990
1160              BNE AGAIN
1170              LDA $FC
1180              CMP 991
1190              BNE AGAIN
1200              LDA TEMPSTORE
1210              STA $01
1220              CLI
1230              RTS
1240 !
1250 !
1260 BLOCKFILL    JSR PARAMETER              !START
1270              LDA $14
1280              STA $FB
1290              LDA $15
1300              STA $FC
1310              JSR PARAMETER              !END
1320              LDA $14
1330              STA 990
1340              LDA $15
1350              STA 991
1360              JSR PARAMETER              !VALUE
1370              LDA $14
1380              STA $FD
1390              LDY #00
1400 AGAIN2       LDA $FD
1410              STA ($FB),Y
1420              LDA $FB
1430              CLC
1440              ADC #1
1450              STA $FB
1460              LDA $FC
1470              ADC #0
1480              STA $FC
1490              LDA $FB
1500              CMP 990
1510              BNE AGAIN2
1520              LDA $FC
1530              CMP 991
1540              BNE AGAIN2
1550              RTS
1560 !
1570 !
1580 CHARMOVE     JSR PARAMETER              !DESTINATION
1590              LDA $14
1600              STA $FB
1610              LDA $15
1620              STA $FC
1630              LDA #00
1640              STA $FD
1650              LDA #$D0
1660              STA $FE
1670              LDA 56334
1680              AND #254
1690              STA 56334
1700              LDA $01
1710              AND #251
1720              STA $01
1730              LDY #0
1740 AGAIN3       LDA ($FD),Y
1750              STA ($FB),Y
1760              CLC
1770              LDA $FB
1780              ADC #1
1790              STA $FB
1800              LDA $FC
1810              ADC #0
1820              STA $FC
1830              CLC
1840              LDA $FD
1850              ADC #1
1860              STA $FD
1870              LDA $FE
1880              ADC #0
1890              STA $FE
1900              LDA $FD
1910              BNE AGAIN3
1920              LDA $FE
1930              CMP #$D8
1940              BNE AGAIN3
1950              LDA $01
1960              ORA #04
1970              STA $01
1980              LDA 56334
1990              ORA #1
2000              STA 56334
2010              RTS
2020 !
2030 !
2040 PARAMETER    JSR CHRCOM
2050              JSR FRMNUM
2060              JSR GETADR
2070              RTS

READY.
```

● **Listing 1 — source code of routines. Only for those who have an assembler program**

```
1 DATA76,15,192,76,59,192,76,94,192,76,211,192,76,20,193,32,212,225,32,110
2 DATA192,165,20,72,165,21,72,32,110,193,166,20,164,21,104,133,21,104,133
3 DATA20,173,232,3,240,4,169,54,133,1,169,20,32,95,225,169,55,133,1,96,32
4 DATA110,193,165,20,133,251,165,21,133,252,160,0,165,1,141,231,3,41,240,120
5 DATA133,1,177,251,141,230,3,173,231,3,133,1,88,96,32,110,193,165,20,133
6 DATA251,165,21,133,252,32,110,193,165,20,141,222,3,165,21,141,223,3,32,110
7 DATA193,165,20,133,253,165,21,133,254,173,229,3,240,8,169,254,141,229,3
8 DATA76,147,192,169,240,141,229,3,141,231,3,45,229,3,120,133,1,160
9 DATA0,177,251,145,253,165,251,24,105,1,133,251,165,252,105,0,133,252,165
10 DATA253,24,105,1,133,253,165,254,105,0,133,254,165,251,205,222,3,208,219
11 DATA165,252,205,223,3,208,212,173,231,3,133,1,88,96,32,110,193,165,20,133
12 DATA251,165,21,133,252,32,110,193,165,20,141,222,3,165,21,141,223,3,32
13 DATA110,193,165,20,133,253,160,0,165,253,145,251,165,251,24,105,1,133,251
14 DATA165,252,105,0,133,252,165,251,205,222,3,208,232,165,252,205,223,3,208
15 DATA225,96,32,110,193,165,20,133,251,165,21,133,252,169,0,133,253,169,208
16 DATA133,254,173,14,220,41,254,141,14,220,165,1,41,251,133,1,160,0,177,253
17 DATA145,251,24,165,251,105,1,133,251,165,252,105,0,133,252,24,165,253,105
18 DATA1,133,253,165,254,105,0,133,254,165,253,208,222,165,254,201,216,208
19 DATA216,165,1,9,4,133,1,173,14,220,9,1,141,14,220,96,32,253,174,32,138
20 DATA173,32,247,183,96
100 FORI=49152 TO 49527
110 READ X: T=T+X: POKEI,X: NEXT
120 IF T <> 49676 THEN PRINT"ERROR IN DATA"
130 REM
140 REM MEMORY MANAGEMENT ROUTINES FOR THE COMMODORE 64
150 REM A E WEBB 1984
```

● **Listing 2 — the same program which you can type in using BASIC**

Smile, then get serious

New games and more serious software for the CBM 64 examined and rated by our experts.

Fire Ant £7.95

Mogul, PO Box 4BT, 35-37 Wardour St, London W1A 4BT

I'm a fast reader, but I was only halfway through the crammed-to-bursting instruction screen when it moved on. There's another chance when the demo has finished but a little longer reading time and better spacing would be nice.

The program, happily, is much better than its original impression. As the sole surviving soldier ant, you must rescue your Queen, held hostage on Screen 8 by Scorpions. Success brings missions to further Scorpion colonies. At first glance, it might seem a Pacman variation — the screens are visible mazes with objects scattered round, patrolling Scorpions lay eggs, and randomly change to purple, double-speed frenzy.

Closer inspection reveals yellow gates and force-fields blocking your way and further hazards become apparent during play. Lightning reflexes help but you must collect the right objects in the right order, placing them in the right places to unblock tunnels and circumvent traps. Objects will kill you if they are taken out of order. And death is not a pretty sight, happening many times before you discover the correct method for each screen-escape and even then, Scorpion stings may get you.

This requires patience, multiple eyes, low cunning and a fast joystick. It's very addictive — dawn broke as I reached level 6. Will satisfy adventure and arcade fans. More please! **D.C.**

instructions	40%
playability	90%
graphics	90%
value for money	85%

★ ★ ★ ★

Turtle Jump £6.99

Romik, 272 Argyll Ave, Slough, Berks

The screen setting is a map of islands in the Caribbean, with turtles swimming between them. Your objective is to get from one island to another and collect treasure points. However, it's not quite as simple as that.

You can jump short distances and travel on anything solid. This means you can travel from one island to another on the backs of the turtles, if you can keep your balance. The only trouble is that if a crocodile appears all the turtles dive. There are a few logs you could usefully jump onto, and there are also some small volcanic islands that appear and later sink beneath the surface. So it is possible, with difficulty, to travel the islands.

Food grows on the islands and is used to top up your energy level. There is an energy barometer displayed on the screen to guide you when it's time for a forage.

You must recover the treasure by jumping in while the chest lid is open and getting out again before it closes. The longer you're in the more treasure you collect. Collected treasure must be taken back to your home base.

A nice game whose theme is different from the run of the mill. I found it difficult to keep on the backs of the turtles and consequently tended to end up on one island far from the treasure but feeding myself silly. Needs joystick. **L.C.**

instructions	70%
playability	60%
graphics	70%
value for money	70%

★ ★ ★

Multisound Synthesizer £14.99

Romik, 272 Argyll Ave, Slough Berks

This utility is designed to allow you to use the sound capabilities of the 64 without POKEing. The range of control offered is enormous and this review can only hint at all the features available.

The synthesizer consists of three screen displays. You set up the characteristics of the note you want using a control screen. This allows you to adjust the attack, decay, sustain, release etc. The levels set for each parameter are indicated by a bar chart display.

Switching to the keyboard screen displays a three octave keyboard with the notes suitably identified, e.g. Q is note C, and 2 is C sharp. As the 64 has four rows of keys the set-up becomes similar to a two keyboard organ. As you play the note in use is indicated. You can move the entire keyboard up or down a few octaves as required. I found this presentation made it very easy to play.

The third screen is for special effects, giving complete control over all the remaining sound features built into the 64. There are just too many to itemise. You have control over all the filtering effects, oscillator and envelope sweeping, ring modulation etc.

You can obviously create tunes, but you can also append tunes, store up to nine tunes in the 64's memory, superimpose tunes over one of eight built-in drum routines, or create your own. Lastly the tunes can be SAVEd to tape or disk.

This program is great for two types of user — the one fingered organists and the experimenter. It is an easy way of trying out different settings so you know what you want to code into your own programs. **L.C.**

instructions	70%
ease of use	70%
display	80%
value for money	80%

★ ★ ★ ★

Cash Controller disc £14.95

Richard Shepherd Software, 23-25 Elmshott La, Cippenham, Berks

A potentially useful package for those who like to keep track of their expenditure. The routines are nicely written and crash-proof.

The package starts by presenting a Main Menu. First-time users would then select from this the Budget Menu and set up headings. You are allowed up to 16, e.g. car, heating, phone etc. There is an option to change these if you have second thoughts. You can then allocate a budget to each. Finally you can transfer to the Bank Account Menu and set up an opening balance. To cut out unnecessary repetition, you can even set up standing orders.

You would subsequently update via the Bank Account Menu. Entering transactions is very simple and prompts are clear. First you enter the date, then a short (up to 10 characters) description. Next you enter the heading to be debited (or credited) and the amount. That's all there is to it.

The computer does the appropriate calculations and updating as necessary. You can then call up a number of reports, to the screen or printer.

You can print out your budget headings, just for reference. More to the point you can print out details of your budgets and variances (the difference between what you allowed and what you spent). You can also print a statement listing all transactions between two specified dates.

I found the package foolproof. My only niggle is that there is no quit option, which means turning the computer off at the end. **L.C.**

instructions	90%
ease of use	80%
display	70%
value for money	80%

★ ★ ★ ★

Find out just how tasty worms can be — just type in Peter Godbehere's program for the unexpanded VIC-20

Peck, peck... but watch out for the spider

Your challenge is to eat the worms which keep appearing in your burrow. However, a spider is after your blood — and his touch is lethal. The controls are included in the game.

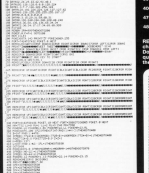
How it works

10-190 define characters
200-250 set up opening screen and wait for keypress
260-330 put burrow on screen
340 set up variables
350 puts man on screen
360-390 put random worms on screen
400-460 check for man hitting monster
470 prints score at the top of the screen
480-520 check for the directional keys to be hit
530-580 moves man left
590-640 moves man right
650-730 moves man down
740-820 moves man up
830-880 moves monster left to right
890-920 moves monster up or down
930-960 add 10 to score and make worms disappear
970-1010 got your routine

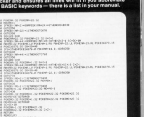
Variables

MA screen position of the top left corner of your man
LU top left character of your man
LL bottom left of your man
RU top right character of your man
RL bottom right of your man
SC score
MO screen position of the top left corner of the monster

REMs have been inserted in the lines above control characters to guide you as you type in Tweet Tweet. Do not enter the REMs. Remember also that it's quicker and ensures all lines will fit if you abbreviate the BASIC keywords — there is a list in your manual.



```
10 DATA0,6,15,59,127,31,15,3
20 DATA0,0,128,128,128,192,96
30 DATA3,1,1,0,0,0,1,7
40 DATA48,24,184,252,124,188,28,128
50 DATA0,0,0,1,1,1,3,6
60 DATA0,96,240,220,254,240,240,192
70 DATA12,24,29,63,62,53,48,1
80 DATA192,128,128,0,0,0,128,224
90 DATA1,3,7,7,126,222,254,124
100 DATA128,192,224,224,126,127,127,62
110 DATA68,238,255,255,255,255,187,17
120 DATA0,0,0,0,0,0,0,0
130 DATA0,3,15,29,31,59,60,31
140 DATA0,192,240,184,248,220,60,248
150 DATA18,36,68,136,136,68,34,34
160 DATA72,36,34,17,17,34,68,68,999
170 F=7168
180 READA:IFA=999THENGOTO200
190 POKEF,A:F=F+1:GOTO180
199 REM [CLR]
200 POKE36879,143:PRINT"■":POKE36869,255
210 FORT=7424TO7432:POKET,0:NEXT
215 REM[CLR][WHT][CRSR DOWN][6CRSR RIGHT][CRSR DOWN][7CRSR LEFT][2RSR DOWN]
220 PRINT"■■■■■■■TWEET TWEET■■■■■■■■V■■■■■■■■P,GODBEHERE":SC=0
225 REM[CRSR DOWN][BLK][RVS ON][7 CRSR RIGHT][2 CRSR DOWN][6 CRSR LEFT]
230 PRINT"■■■■■■■■Z=LEFT■■■■■■■■X=RIGHT■■■■■■■■■=UP■■■■■■■=DOWN"
235 REM[CRSR DOWN][CRSR RIGHT][WHT][RVS ON]
240 PRINT"■■■HIT ANY KEY TO START"
250 POKE198,0:WAIT198,1
255 REM[CLR][BLK][CRSR DOWN][20 CRSR RIGHT][20 CRSR RIGHT]
260 PRINT"■■UJJJJJJJJJJJJJJJJJJJJJJJJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■U
"
265 REM[CRSR UP][WHT][CRSR RIGHT][BLK][WHT][BLK][CSR R[20CRSR RIGHT][20CRSR RIGH
T]
270 PRINT"UJJ■■■JJJJJJJJJJ■■■JJJJJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■
JJ"
275 REM[CRSR UP][WHT][BLK][RIGHT][WHT][BLK][CSR RGHT][20CRSR RIGHT][20CRSR RIGH
T]
280 PRINT"UJJJJJJJJ■■■JJJJJJJJ■■■JJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■
JJ"
285 REM[CRSR UP][WHT][BLK][RIGHT][WHT][BLK][CSR RGHT][20CRSR RIGHT][20CRSRRIGHT
290 PRINT"UJ■■■JJJJJJJJJJJ■■■JJJJJJJJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■
JJ"
295 REM[CRSR UP][WHT][BLK][RIGHT][WHT][BLK][CSR RGHT][20CRSR RIGHT][20CRSRRIGHT
300 PRINT"UJJJJJ■■■JJJJJJJJJJJ■■■JJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■
JJ"
305 REM[CRSR UP][WHT][BLK][RIGHT][WHT][BLK][CSR RGHT][20CRSR RIGHT][20CRSRRIGHT
310 PRINT"UJJJJJJJJJ■■■JJJJ■■■JJJJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■
JJ"
315 REM[CRSR UP][WHT][BLK][RIGHT][WHT][BLK][CSR RGHT][20CRSR RIGHT][20CRSRRIGHT
320 PRINT"UJJJ■■■JJJJJJJ■■■JJJJJJJ■■■■■■■■■■■■■■■■■■■■UJ■■■■■■■■■■■■■■■■■■■■
J"
330 FORT=8163TO8186:POKET,10:NEXT:FORT=38883TO38905:POKET,0:NEXT
340 MA=7732:LU=0:RU=1:LL=2:RL=3:Z=0:MO=7930
350 POKEMA,LU:POKEMA+1,RU:POKEMA+22,LL:POKEMA+23,RL
360 POKE36876,200:IFZ>9THENIFINT(RND(1)*4)<>1THENGOTO400
370 B=INT(RND(1)*470)
380 IFPEEK(7724+B)=10ORPEEK(7725+B)=10ORPEEK(7724+B)=11THENGOTO400
390 POKE7724+B,8:POKE7725+B,9:Z=Z+1
400 K=PEEK(197)
410 L=INT(RND(1)*2):IFL=1THENGOTO830
420 GOTO860
430 POKE36876,0:IFMA=MOORMA+1=MOORMA-1=MOTHENGOTO970
440 IFPEEK(MO-22)=11THENGOTO890
450 IFPEEK(MO+44)=11THENGOTO910
460 POKEMO,12:POKEMO+1,13:POKEMO+22,14:POKEMO+23,15
465 REM[HOME][RVS ON][ORN]
470 PRINT"■■SCORE=";SC
480 IFK=33THENGOTO530
490 IFK=26THENGOTO590
500 IFK=12THENGOTO650
510 IFK=36THENGOTO740
520 GOTO350
530 IFPEEK(MA-1)=10THENGOTO350
540 LU=0:RU=1:LL=2:RL=3
550 POKEMA+1,32:POKEMA+23,32
560 MA=MA-1
570 IFPEEK(MA-1)=9ORPEEK(MA+21)=9THENGOSUB930
580 GOTO350
590 IFPEEK(MA+2)=10THENGOTO350
600 LU=4:RU=5:LL=6:RL=7
```



```
610 POKEMA,32:POKEMA+22,32
620 MA=MA+1
630 IFPEEK(MA+2)=8ORPEEK(MA+24)=8THENGOSUB930
640 GOTO350
650 IFPEEK(MA-22)=11THENGOTO670
660 GOTO350
670 SO=128:D=0
680 POKEMA+22,32:POKEMA+23,32:D=D+1
690 IFPEEK(MA-44)=8ORPEEK(MA-45)=8THENZ=Z-1:SC=SC+10
700 MA=MA-22:POKEMA,LU:POKEMA+1,RU:POKEMA+22,LL:POKEMA+23,RL:POKE36878,15
710 POKE36876,SO:SO=SO+40
720 IFD=3THENPOKE36876,0:POKEMA+44,11:GOTO350
730 GOTO680
740 IFPEEK(MA+44)=11THENGOTO760
750 GOTO350
760 SO=200:D=0
770 POKEMA,32:POKEMA+1,32:D=D+1
780 IFPEEK(MA+44)=8ORPEEK(MA+45)=9THENSC=SC+10:Z=Z-1
790 MA=MA+22:POKEMA,LU:POKEMA+1,RU:POKEMA+22,LL:POKEMA+23,RL:POKE36878,15
800 POKE36876,SO:SO=SO-40
810 IFD=3THENPOKE36876,0:POKEMA-22,11:GOTO350
820 GOTO770
830 IFPEEK(MO+2)<>32THENGOTO430
840 POKEMO,32:POKEMO+22,32:MO=MO+1
850 GOTO430
860 IFPEEK(MO-1)<>32THENGOTO430
870 POKEMO+1,32:POKEMO+23,32:MO=MO-1
880 GOTO430
890 POKEMO,32:POKEMO+1,32:POKEMO+22,32:POKEMO+23,32
900 MO=MO-66:GOTO460
910 POKEMO,32:POKEMO+1,32:POKEMO+22,32:POKEMO+23,32
920 MO=MO+66:GOTO460
930 SC=SC+10
940 POKEMA-1,32:POKEMA+21,32
950 POKEMA+2,32:POKEMA+24,32:Z=Z-1
960 RETURN
965 REM[CLR]
970 PRINT"■":POKE36878,0
975 REM[WHT][RVS ON][2 CRSR DOWN][5 CRS RIGHT]
980 PRINT"■■■■■■■■■GOT YOU!"
985 REM[CRSR DOWN][3 CRSR RIGHT][RVS ON]
990 PRINT"■■■■YOU SCORED";SC
1000 FORT=0TO1000:NEXT
1010 GOTO220
```

Audiogenic Vic-20 Games...

...never turn your back on them.

Spiders of Mars, Outworld and Cloudburst - three classic cartridge games - now available on cassette! Bonzo and Kaktus, both bestsellers, are now joined by the latest crazy mazey game - Shifty! From all major software shops everywhere.

These incredible games cassettes all require 8K minimum RAM expansion. All prices include V.A.T. and P. & P.

WRITE OR PHONE FOR FREE COLOUR CATALOGUES.

Bonzo	£7.95
Kaktus	£7.95
Shifty	£5.95
Spiders of Mars	£5.95
Outworld	£5.95
Cloudburst	£5.95

Audiogenic LTD

BONZO
KAKTUS
Shifty
SPIDERS OF MARS
OUTWORLD
CLOUDBURST

Audiogenic, PO Box 88, Reading, Berks, England. Tel: (0734) 586334.

SUPERSOFT

the name to remember

for games

For only £8.95 you can buy a game that's exciting, soothing, and frustratingly addictive – all at the same time! STIX looks so different and sounds so different from all those other games that it will seem like being in another dimension when you sit down to play.

CRAZY KONG £8.95

WILDFIRE £6.95

HALLS OF DEATH £8.95

for business

Show your computer who's master with BUSICALC! Spreadsheet programs are used by large and small businesses to juggle with figures, prepare reports and so on. Some are very powerful indeed. The problem is that they're difficult to learn, and tricky to use – which is why we came up with the BUSICALC series.

Whether you choose BUSICALC 1, BUSICALC 2, or BUSICALC 3 you'll get a program you can understand – and one that almost seems to understand you. Use it in the home, use it for teaching, use it at work – it'll save you time and money.

for programmers

MIKRO is a full 6502/6510 ASSEMBLER with the power that professional programmers need, yet so simple to use that we recommend it to beginners! The MIKRO cartridge has many other facilities including editing commands and a machine language monitor, all for £57.50.

There's much more for the 64 in the SUPERSOFT catalogue. Ask your computer dealer for a copy, or phone 01-861 1166.

The Best 64 Software

SUPERSOFT, Winchester House, Canning Road, Wealdstone, Harrow, Middlesex, HA3 7SJ Telephone: 01-861 1166